Remote Rendering Using Vtk and Vic

Robert Olson Michael E. Papka

Futures Laboratory
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439
papka@mcs.anl.gov

1 Introduction

This paper presents a remote rendering application that involves the extension of the Visualization Toolkit (vtk) and the Video Conferencing Tool (vic) for use in remote rendering complete with interaction from the remote site using the vic user interface. Vtk is an open source C++ Library, with Tcl, Python, and Java bindings for computer graphics, image processing, and visualization [3]. Vtk provides a higher level of support beyond the traditional low-level libraries, for creating visualization applications. Vtk includes algorithms to support the visualization of scalars, vectors, and tensors. Vic is a flexible tool built by Lawerence Berkeley Laboratory for real-time video conferencing over the Internet [2]. Vic's user interface is built as Tcl/Tk script embedded in the applications. This allows for developers to prototype changes to the interface in a simple and straightforward manner.

2 Remote Rendering

The vic source code was modified to stream output from interaction with its window to a given port on a given machine. This stream was then received by a standard vtk application that was augmented with a network-based interactor. This allowed multiple sites to interact using the standard vtk interactor controls with the remote vtk application via the vic window. Figure 1 shows a diagram of the system. The output of the vtk application at this point is scan converted from the server machine and streamed into a standard video capture card for broadcast on a multicast address. This application allowed for a remote user to have direct control of the application without concern about the rendering power of the host machine. The user was able to interact in real-time with a one million-polygon model from a standard desktop PC. Figure 2 shows a snapshot of the remote rendering output and the output as seen in the vic window.

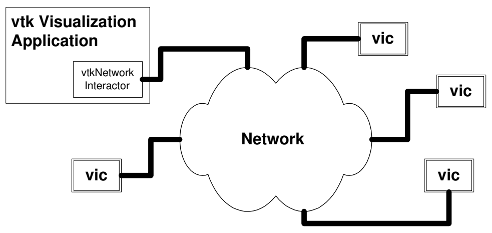

Figure 1: System diagram of vic-vtk system.

Users need only to apply a set of patches to the vic source code and recompile for the remote site. Users with existing vtk applications need only to replace their current vtkInteractor with a vtkNetworkInteractor to address the remote user interaction. As mentioned before, the current implementation requires the scan conversion of output for video streaming, but a new version is under way that would build a new vtkRenderer for generating the network video stream. This will be based on ANL's CAVEav library that generates synthetic video streams from within a CAVE application for multicast broadcast [1] and the vic tool.

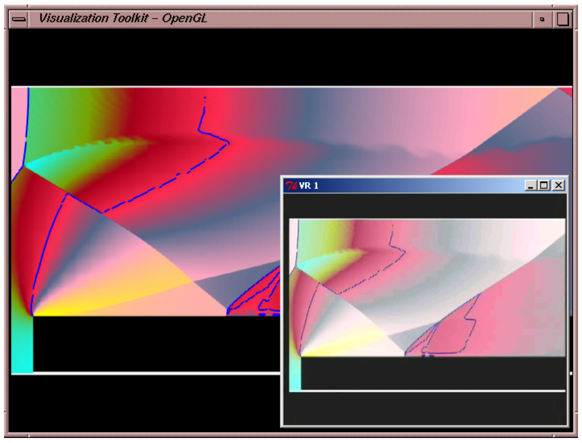

Figure 2: Output of the vtk server, with an inset image of the vic client window. Relative size is maintained, as well as the difference in colormaps.

3 Conclusion

This system provides a large community of users with the capability of remote visualization, with interactivity at the remote site using existing tools and infrastructure. The extensions to vtk will be made available to the community, allowing any developer the ability to plug in the networked-based interactor to an existing or future vtk applications. In addition, since the source code will be made available, developers will have the opportunity to extend the functionality. We have begun to develop the networked rendering side allowing for users to use this tool wherever they have the most graphics horsepower and connect to other users for remote viewing without the need for the scan conversion.

4 Acknowledgements

Special thanks to Tony Lavoie for the additions and improvements he has made to the initial prototype. Additionally we would like to thank the members of the Access Grid community that have field tested the system. This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

References

[1] Terrence L. Disz, Michael E. Papka, and Rick Stevens. Ubiworld: An environment integrating virtual reality, supercomputing, and design. In *Proceedings of the Heterogenous Computing Workshop*, Geneva, Switzerland, April 1997.

[2] Steven McCanne and Van Jacobson. Vic: A flexible framework for packet video. In *Proceedings of the Third ACM International Multimedia Conference '95*, pages 511–522. ACM, ACM Press, November 1995.

[3] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphic*. Prentice Hall, 1995.